(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 April 2001 (19.04.2001)

PCT

(10) International Publication Number
WO 01/27880 A1

(51) International Patent Classification[7]: G06T 15/70

(21) International Application Number: PCT/US00/27696

(22) International Filing Date: 6 October 2000 (06.10.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/414,883 8 October 1999 (08.10.1999) US

(71) Applicant: NINTENDO SOFTWARE TECHNOLOGY CORPORATION [US/US]; 5001 150th Avenue N.E., Redmond, WA 98052 (US).

(72) Inventors: COMAIR, Claude; 5275 Granville Street, Vancouver, British Columbia V6M 3B9 (CA). GHALI, Prasanna; 1788 W. Georgia Street, Suite #101, Vancouver, British Columbia V6G 2V7 (CA). SAMRA, Samir, Abou; 2677 West 35th Avenue, Vancouver, British Columbia, V6N 2L9 (CA). FAM, Sun, Tjen; #1105-5183 Melbourne Street, Vancouver, British Columbia V5R 6E6 (CA). LI, Xin; 4008 258th Way S.E., Issaquah, WA 98029 (US).

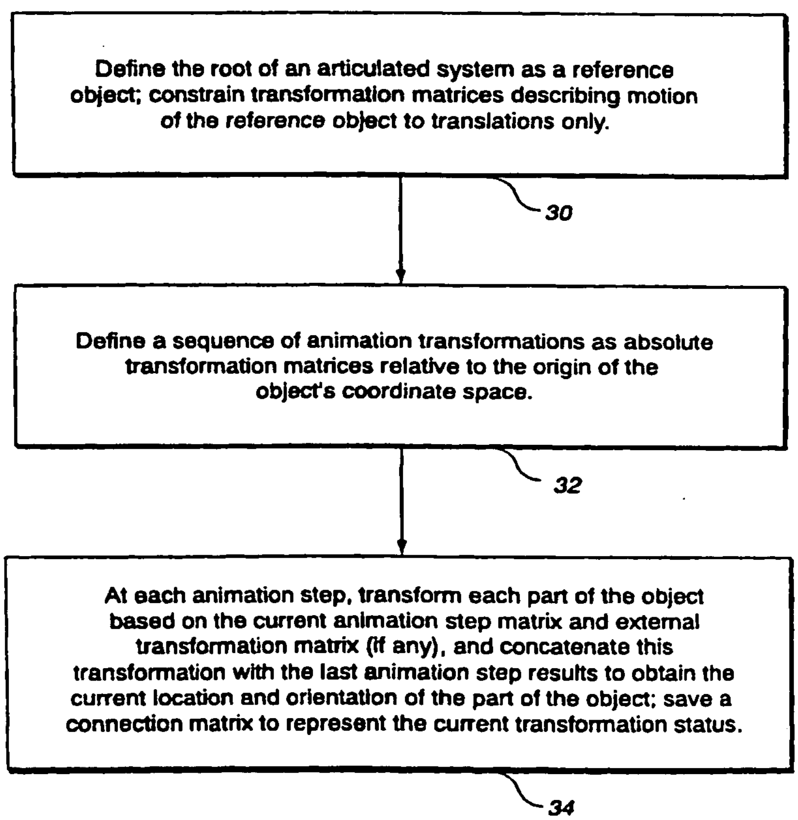
(74) Agent: FARIS, Robert, W.; Nixon & Vanderhye P.C., Suite 800, 1100 North Glebe Road, Arlington, VA 22201-4714 (US).

(81) Designated States (*national*): AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR INSERTING EXTERNAL TRANSFORMATIONS INTO COMPUTER ANIMATIONS



(57) Abstract: Transformation matrices describing the motion of an articulated reference object (50) specify translations of the reference object (50) and/or its articulated parts, and are relative to the origin of the object's coordinate system (22). During the real-time animation process, a connection matrix is saved to represent the current transformation status of the animated object (50). This connection matrix being obtained by continuously concatenating an external transformation (if any) with the next animation step. This process allows efficient insertion of external transformations (e.g., resulting from operating real-time interactive user controls). Use of absolute transformations to develop the connection matrix allows a first animation sequence to be interrupted at any arbitrary point and immediately, smoothly followed by a second animation sequence (e.g., walking can be interrupted at any time any turned into running, jumping, etc.).

WO 01/27880 A1

patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *With international search report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD AND APPARATUS FOR INSERTING EXTERNAL TRANSFORMATIONS INTO COMPUTER ANIMATIONS

FIELD OF THE INVENTION

This invention relates to the field of computer graphics, and more specifically, to modeling and/or simulating the behavior and motion of real-world articulated objects such as humans, animals and birds in real-time 3-dimensional video games and other interactive contexts. Still more particularly, this invention relates to a unique solution to the problem of efficiently inserting external matrix transformations (e.g., generated in real time) to computer-modeled animations and/or changing animation sequences at an arbitrary point.

BACKGROUND AND SUMMARY OF THE INVENTION

Computers and video games have revolutionized animation—the process of showing a sequence of images in rapid succession to create the illusion of motion. In the past, animations were created by a human artist drawing each of the many still frames required (e.g., over 1000 frames for each minute of animation). Now computers can do most of this work.

For example, computers are commonly used to automatically generate an arbitrarily-long sequence of successive incremental frames between two "key frames" specified by the animation artist or captured by an image sensing device. Using this technique, a first frame and a last frame of an animation sequence are specified, along with the duration of the sequence. Based on this information, the computer automatically generates the many intermediate frames between the first and last frames using a process called interpolation. Figure 1 illustrates this well

known technique. In the Figure 1 example, the animation artist specifies a first key frame 10 in which the arm of character 11 is lowered, and a second key frame 12 in which character 11's arm is raised. The computer interpolates to automatically generate intermediate frames 14(1), 14(2), 14(3), ..., 14(N), showing character 11's arm in the various intermediate positions between the lowered and raised positions.

Pre-storing each still picture in its entirety would take up too much space. Therefore, the computer uses a sequence of mathematical transformations to specify incremental changes between the successive frames of the animation. These mathematical transformations can be conveniently represented by a mathematical construct called a transformation matrix ("M") that specifies the amount of translation, rotation and scaling in each of the three possible axes (x, y, z) of three-dimensional space. Such transformation matrices are commonly used in computer graphics to represent changes in position, orientation and/or size between successive images.

Figure 1 shows how a first transformation matrix M 16(1) transforms the beginning key frame 10 to a first intermediate frame 14(1); a second transformation matrix M 16(2) transforms the first intermediate frame 14(1) to a second intermediate frame 14(2); and so on, with a last transformation matrix M 16(N) transforming the last interpolated intermediate frame 14(N-1) to the second key frame 12.

In this example, the computer models character 11 just as real humans or animals are constructed—out of different parts (e.g., legs, arms and a torso) interconnected by articulated joints. See Figure 1A. This modelling technique allows the computer to break up the animation problem into more easily handled

parts (for example, how the character's right arm moves relative to the character's torso; and how the character's torso moves relative to the character's surroundings). Dividing a character 11 up into different articulated portions allows the computer to use different transformation matrices to move parts of the character relative to other parts of the character. For example, one transformation matrix may be applicable to the entire character (e.g., to move the entire character relative to a common three-dimensional "world" space). Other matrices may apply to various body parts of the character. Such articulated portions are sometimes represented in a hierarchical structure an example of which shown in Figure 1B. To move a part of the character relative to the rest of the character (e.g., to cause the character to raise its arm), a transformation matrix is applied to that part of the character relative to the rest of the character.

Because the animation must run at high speed (i.e., over a thousand pictures a minute) to create the illusion of fluid motion, it is desirable to pre-compute and pre-store the necessary animation transformation matrices when the animation sequence is authored. Then, during video game play or other presentation of the animation, the computer reads these pre-stored matrices and applies them to achieve a desired visual animation effect.

However, in an interactive situation like a video game, how a character moves and how a scene changes may depend in part on events occurring in real time (for example, a game player's operation of joysticks, buttons or other controls). This means that the real time interactive animation or simulation cannot rely entirely on pre-computed animation matrices, but must instead provide some way for the animation to also be affected by real world inputs. For example, while a character's arm and leg motion may be characteristic of the particular character

and may be essentially the same irrespective of the direction in which the character walks, the direction in which the character walks may be controlled by operation of a keyboard, handheld controller or joystick.

While it would be desirable to use the same pre-computed animation (matrix) sequences for any arbitrary walking direction, this has sometimes been difficult to achieve in the past because of the way the animation transformation matrices are typically pre-computed as representing the incremental change(s) (Δ) between successive frames of the animation. This has in some cases meant that the animation artist at authoring time must anticipate each of the various directions the character will be required to walk, and pre-compute a different animation sequence for each different direction. For example, one technique pre-computes a walking animation sequence for each 10 degrees of the compass—allowing a character to walk in any selectable one of thirty-six different directions. Storing all of these various pre-computed sequences takes up a lot of space in memory, and also does not actually cover every possible walking direction (for example, in this scenario, the video game player cannot control the character to walk in a direction of 245 degrees, but is limited to 240 degrees or 250 degrees). It would be desirable to easily and efficiently insert, into a pre-computed animation sequence, an arbitrary external transformation generated in real time response to user interaction in order to, for example, control the character's orientation and/or translation direction and speed.

Another problem relating to pre-constructed animation matrices for real time interactive animations is how to handle a transition from one animation sequence to another. Past techniques defining each incremental frame of an animation sequence relative to the last frame of the sequence can lead to

unnatural-looking transitions between different animation sequences. Suppose for example that during an interactive game, the game player operates a hand controller to cause character 11 to walk for a time and then suddenly pushes a button to cause the character to run. In many prior systems, it was necessary to finish the walking animation sequence before starting the running animation sequence. This is why some prior games noticeably delay transitions between different animation sequences. One approach to solving this problem is to make each animation sequence very short so the player won't notice any significant delay. In contrast, however, a real world person or animal can immediately change direction or orientation without waiting for a prior motion to complete. It would be desirable to simulate such fluid continuity by allowing an immediate transition from any arbitrary frame of an animation sequence to another animation sequence.

The present invention solves these problems by providing a unique framework for allowing an externally-supplied transformation matrix to further transform an ongoing animation, and for smoothly concatenating animations together seamlessly and arbitrarily. The algorithm can be built in a game-authoring environment that allows artists, musicians and developers to maximize their creative freedom and reduce engineering costs. Techniques provided in accordance with the present invention can be used in the context of a hierarchical articulated system, to efficiently provide lifelike and natural motion of a variety a complex humanoid, animal and other simulated animated objects.

In accordance with one aspect of this invention, a reference object is defined as the root of the articulated system. Transformation matrices describing the motion of this reference object are constrained to translations. This means the

matrices can be easily inverted. The ability to easily invert theses matrices avoids the cost of computing, at run time, a "relative" matrix representing change relative to the last transformation.

In accordance with another aspect of this invention, a sequence of transformation matrices defines an animation sequence for an articulated part of the reference object. This sequence of transformation matrices is constructed as absolute transformations. That is, the translations described by the transformation matrices are not relative to the previous position of the articulated system, but instead are defined with respect to the origin of the object's coordinate system. Because each animation step references the origin (as opposed to a "last" position), each step is essentially self-contained—and can thus serve as the departure point for any other arbitrary transformation (e.g., the first step of a different animation sequence).

In accordance with a further aspect of the invention, a connection matrix is saved at each animation step to represent the current transformation status of the animated object. This connection matrix embodies the result of the entire history of all previous transformations on the object. This connection matrix is obtained at each animation step by continuously concatenating an external matrix, if any, with the transformation matrix for the animation step. The historical information the connection matrix stores is used to further transform the animation, and is also stored for use in the next successive animation step(s).

Although the transformations performed in accordance with the present invention can be implemented by multiplying the various matrices together to achieve a single transformation matrix which is then applied to the object, one

way to think about the transformation process in accordance with one embodiment of the present invention is as follows:

- the object is transformed by the inverse translation of the current animation step;
- the external transformation (if any) is inserted (e.g., based on real time input);
- the object is (re)translated based on the current animation step's translation matrix;
- the connection matrix saved from the last animation step is used to concatenate with above transformation with matrix; and
- the next animation step's transformation matrix is applied to obtain the object's new location and orientation.

Because these techniques relate the object back to its reference coordinate system origin, they can be efficiently used at any arbitrary point in an animation sequence. This allows any arbitrary frame of an animation sequence to be a departure point for a different animation sequence. Furthermore, because the transformation matrices are defined as absolute translations, the transformation calculations are very efficient (for example, perhaps ten times faster than otherwise in some contexts); and provide a very high degree of natural realism and fluid motion. As one dramatic example, a character that is animated in accordance with the present invention can efficiently and rapidly shift from doing backward handstands to doing forward handstands in a very realistic and fluid manner without interruptions. Furthermore, the calculations provided by the present invention are very inexpensive from a computational standpoint, and are therefore particularly suited for high-speed execution on restricted- resource graphics

platforms such as 3D home video game consoles. Such efficiencies also help provide quick prototyping of newly authored animations.

The present invention thus allows an artist to choose a single translation direction (e.g., the most convenient one to model) relative to a reference origin, for authoring a particular animation. The direction of translation and the orientation of the object being animated can be easily and efficiently changed at run-time to any arbitrary orientation and/or translation direction, based on insertion of an external transformation matrix. Furthermore, a connection matrix continuously concatenated with the animation sequence animation matrices can be used to supply a different object location (e.g., based on the object's prior history in the 3D universe). Still further, the artist may author an animation sequence having any arbitrary duration, since the sequence can be efficiently interrupted at any time in the real-time animation process to bridge into a different animation sequence. These added flexibilities make it possible for a graphics artist without much or any programming experience to easily author extremely realistic and fluid animations.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other features and advantages of the present invention will be better and more completely understood by referring to the following detailed description of presently preferred example embodiments in conjunction with the drawings, of which:

Figure 1 shows an example prior art "key frame" animation technique;

Figure 1A shows an example prior art hierarchical articulated character model;

Figure 1B is a schematic illustration of an example prior art database representation of the Figure 1A hierarchical model;

Figure 1C is an example 3D graphics display system the preferred embodiment of the present invention can use to present animations on;

Figure 2 schematically illustrates a reference object in accordance with the present invention;

Figure 2A schematically shows successive transformations of the Figure 2 reference object for sequential animation frames;

Figures 2B and 2C illustrate examples of how any arbitrary animation frame can be a departure for a new animation sequence;

Figure 3 schematically illustrates an example animation process in accordance with the present invention;

Figure 3A schematically shows insertion of an external transformation between frames k and k+1 of an example animation sequence;

Figure 4 shows an example transformation technique for inserting an external rotation between the k and k+1 steps of an animation sequence;

Figures 4A-4D schematically illustrate a step-by-step explanation of the Figure 4 process;

Figure 5 is a flowchart of an example process performed in accordance with the invention; and

Figures 6A and 6B together show an example hierarchical object model provided by the present invention.

DETAILED DESCRIPTION OF PRESENTLY PREFERRED EXAMPLE EMBODIMENTS

Figure 1C shows an example 3-D real time computer animation system 50 that may be used to provide realistic interactive real time 3D simulation and animation in accordance with the present invention. The Figure 1 example system 50 includes a NINTENDO 64® 3-D video game console 52 and associated hand controllers 54a, 54b. A cartridge 56, optical disk or other storage medium storing a software animation (video game) program is operatively connected to console 52. The console 52 is connected to a display device 58 such as a conventional home color television set or computer monitor. Console 52 includes a 3D graphics engine that can render 3D animation on display 58 in real time response to user manipulation of controllers 54a, 54b. The software within cartridge 56 controls console 52 to display a sequence of animated video frames on display 58. These animated video frames can serve to animate an arbitrary three-dimensional character 11 such as the character schematically shown in Figure 2.

Figure 2 shows definition of a reference object 50 with respect to arbitrary three-dimensional character 11. The reference object 50 (which may be a cube) is defined at the root of the hierarchical articulated system used to represent character 11 (see Figures 1B, 7A & 7B for example).

In accordance with one aspect of the present invention, all transformation matrices describing the motion of reference object 50 are constrained to translations only. Thus, these various translations will result in translating the reference object 50 along a piecewise-linear path. Those skilled in the art will recognize that this form of transformation matrix is relatively easy to invert (e.g.,

using only three additions) -- making the inversion process computationally inexpensive and very efficient.

In accordance with another aspect provided by the invention, the transformation matrices defining animations are constructed as absolute transformations. That is, the transformations described by the transformation matrices are not relative to the previous position of the articulated system (e.g., a "Δ" between successive displayed frames), but instead are defined with respect to the origin of the object's coordinate system. Because these matrices are defined with respect to the origin of modeling space for the object, the transformation matrices applied to each animation step all relate back to a common reference origin. One implication of this technique is that any arbitrary animation transformation can follow any other animation transformation. This gives the animation system a much more lifelike and realistic appearance, allowing characters for example to change from a walk to a run or vice versa within the time of a single animation frame.

Figures 2A, 2B and 2C illustrate some the advantages provided by using such absolute transformation matrices. Figure 2A shows an example animation sequence comprising animation step transformations 20(0), 20(1), 20(2), 20(3), 20(4), relative to the modeling coordinate system origin 22. For purposes of simplified illustration, Figures 2A and 2B show the modeling coordinate system as two-dimensional, but those skilled in the art will understand that in a 3-D graphics system this coordinate system will have another axis to make it be three-dimensional rather than two-dimensional.

Each of the various translation transformations of Figure 2A independently relates back to and references the modeling coordinate system origin 24. The

corollary of this construct is that each transformation describes the current position of the reference object 50 without reference to any prior transformation. This powerful construct allows any transformation to be a departure point for any other transformation (e.g., to change to direction of character motion, or even to being a completely different animation sequence).

Figure 2C graphically illustrates this. Suppose axis 26 of Figure 2C represents, in linear form, an animation path defined by the first few frames 20(0), 20(1), ... of an example "walk" animation sequence. Figure 2C shows that any of these frames can be a departure point for an entirely different animation sequence (e.g., a "run" animation sequence). Because each of the transformations in the Figure 2C "walk" animation sequence is absolute (i.e. in accordance with present invention, each change in position (Δ) relative to the last animation frame is represented relative to the object's reference coordinate system origin instead of with respect to the last frame position), it becomes very easy and efficient to "interrupt" one animation sequence at any point in the sequence to begin a different animation sequence.

EXAMPLE COMPUTER PROCESSING STEPS

Figure 3 shows the overall steps system 50 may perform to provide animation in accordance with a preferred example embodiment of the present invention. An initial step 30 defines the root of an articulated system (e.g., the Figure 1A character) as a reference object (see Figure 2), and constrains transformation matrices describing motion of the reference object to translations only. A second step 32 defines a sequence of transformation matrices M_1 ..., M_{n-1} and M_n representing an animation sequence of the object or an articulated portion

of the object, where each matrix is defined with respect to the origin of the object's coordinate system origin. A further step 34 performed at each animation step transforms the object or articulated portion thereof, based on the transformation matrix for the current animation step and an external transformation matrix (if any).

At any arbitrary step k of the animation, such an external rotation matrix M_e (e.g., reflecting real time interactive inputs from a human game player) may be inserted between steps k and k+1 so the sequence would be changed to reflect the influence of this external transformation. The external rotation matrix may be a 4x4 matrix defining rotation, translation and/or scaling in any of the x, y and z directions. The transformation(s) defined by the external matrix may be supplied in whole or in part in response to interactive realtime input from a human player via a user input device; or they may be supplied by other events (e.g., operation of a concurrently running program). Step 34 concatenates this transformation with the last animation step results to obtain the current location and orientation of the object. Step 34 also saves, at each step in the animation, a connection matrix to represent the current transformation status of the animated object for use in such concatenation -- and for use in carrying forward to succeeding animation steps, the cumulative distilled history of the orientation, size and position of the object or articulated portion thereof.

Figure 3A graphically illustrates the transformation procedure of block 34. In the Figure 3A graphical depiction:

- <OX, OY> shows the position and orientation of an arbitrary object after a last animation step k;

- < OX', OY' > shows the consequence after k+1 steps without the interference of the external rotation; and
- < OX'', OY' '> is the result with the external rotation inserted for the k+1 step.

Figure 3A uses the following notations:

- M_k and M_{k+1} are "absolute" transformation matrices of animation step k and k+1, which are defined with respect to the origin of the object's coordinate system;
- $M_{k+1} = \Delta M_{k+1} M_k$ where ΔM_{k+1} is the k+1 step transformation matrix defined with respect to the previous transformation M_k; and
- M_e is the external transformation matrix containing only rotations.

An intuitive method might be to push the external transformation matrix M_e onto the stack after pushing the current transformation M_k, then adding the "relative" transformation matrix M_{k+1} on top of that. This can be described by equation:

$$M''_{k+1} = \Delta M_{k+1} M^r_e M_k$$

In practice, since all matrices are stored as the "absolute" transformations, it would be expensive to compute the relative transformation ΔM_{k-1} at runtime. Our approach instead uses the formula:

$$M''_{k+1} = M_{k+1}(-M^t_k) M^r_e M^t_k M_k$$

As a practical matter, this calculation can be performed by multiplying all of the various transformation matrices together to produce a resultant matrix that is then used to transform the object or articulated segment of the object.

Figure 4 shows a more detailed sequence of computer-performed steps that may be used to efficiently implement the Figure 3A transformation, and Figures 4A-4D graphically illustrate these various steps. In each of figures 4A-4D, < OX, OY > represents the object location and orientation before the transformation operation, while < OX', OY' > indicates that after the transformation. The first step is to invert the matrix M_k for the current animation step, and to transform the object by the resulting inverse transformation (translation) of matrix M_k for the current animation step (block 40) (see Figure 4B). Next, an external transformation matrix M_e is inserted if one is being used (block 42) (see Figure 4C). This external transformation matrix commonly may include orientation transformations based on real-time user inputs such as operation of a joystick or the like, but may also or alternatively include other transformations such as translation and/or scaling.

The object is then re-translated with the current step's animation translation matrix M_k (block 44) (see Figure 4D) -- thus "undoing" the Figure 4B transformation by the inverse matrix. The result is concatenated with a connection matrix saved from the last animation step (block 46). The next animation step's transformation matrix M_{k-1} is applied to the concatenated result (see Figure 4A -- which operation may be performed at the beginning or at the end of the concatenation process as desired) to obtain the object's location and orientation. It is easy to observe that the final result of the transformation shown in Figure 4D coincides with the one shown in Figure 3A.

Figure 5 is a flowchart of an example process performed by the Figure 1C system to animate character 11. In this example, system 50 initially uses an identity matrix as the connection transformation matrix (block 102), and pushes

the connection transformation matrix onto a system transformation matrix stack (block 104). System 50 then determines whether there is an external transformation (e.g., supplied based on operation of controller(s) 54 by a human user) (block 106). If there is an external transformation ("yes" exit to decision block 106), system 50 pushes the current translation matrix onto the stack (block 108); then pushes external transformation matrix onto the stack (block 110); and finally, pushes the inverse of the current translation matrix onto the stack (block 112). Thus, the external transformation matrix is effectively concatenated with the current transformation matrix and the inverse of the current transformation matrix (see Figures 4B-4D).

Whether or not there is an external transformation, system 50 pushes the transformation matrix for the next animation step onto the stack (block 114), multiplies all matrices onto the stack to produce a current connection matrix (block 116), and saves the result (block 118). This resulting current connection matrix is used to generate display of the k+1th animation frame, and is also pushed onto the stack for use in the k+2d animation step (block 104). As will be understood by those skilled in the art, further conventional transformations (e.g., from object space into world space and/or camera space) may be applied before the animation-transformed object is displayed on display device 58.

Figures 6A-6B show an example hierarchical data structure 200 that can be used to represent an arbitrary object 11 in accordance with the present invention. Data structure 200 is conventional except for its transformation table shown in Figure 2B -- which in this example stores sequences of transformation matrices of the type discussed above. In one example, each articulated joint of the 3D object will be separately defined in hierarchical data structure 200, and will have its own

respective sequence of animation transformation matrices for each of N different animations (e.g., running, walking, doing handstands, falling, etc.) The matrices represented in transformation table 202 may be 4x4 matrices to allow translation, rotation and scaling factors to be represented in the same matrix. The transformation matrix sequences shown in Figure 6B are used to perform the animation processes described herein.

* * * * *

While the invention has been described in connection with what is presently considered to be the most practical and preferred embodiments, it is to be understood that the invention is not to be limited to the disclosed embodiments, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the scope of the appended claims.

WHAT IS CLAIMED IS:

1. In a videographics system including at least one interactive user control input, a method of animating a three-dimensional display object comprising:

(a) determining an external transformation based at least in part on said interactive user control inputs;

(b) multiplying said external transformation matrix by a current transformation matrix and its inverse to produce a resultant matrix; and

(c) displaying said next animation frame including said display object based at least in part on said resultant matrix.

2. A method as in claim 1 wherein said generating step (b) comprises constraining said next animation frame transformation matrix to specify translation only.

3. A method as in claim 2 wherein said object is defined relative to an object coordinate system having an origin and includes at least one movable portion that is movable relative to a reference portion, and said generating step (b) comprises generating a connection matrix by concatenating a next animation frame transformation matrix with said external matrix for said movable portion relative to said object coordinate system origin.

4. A method as in claim 1 wherein said display object is modeled as a hierarchical system of articulated parts, and said step (b) is performed for at least some of said parts.

5. A method as in claim 1 wherein said object is modeled as an articulated system having a root defined by a reference object, and said steps (a) and (b) are performed for said reference object.

6. A method as in claim 1 wherein said step (b) is performed for each step of said animation.

7. A method as in claim 1 wherein said step (b) comprises continuously concatenating the external transformation with a next animation step to produce a connection matrix.

8. A method as in claim 1 further including concatenating the resultant of step (b) with a connection matrix saved from a just previous animation step.

9. A method as in claim 1 wherein said external transformation matrix contains only rotations.

10. A method as in claim 1 wherein step (b) is performed by calculating

$$M''_{k+1} = M_{k+1}(-M^t_k)\, M^r_e\, M^t_k\, M_k$$

where M_k and M_{k+1} are absolute transformation matrices of animation steps, k and k+1 defined with respect to the origin of the object's coordinate system; $M_{k-1} = \Delta M_{k+1} M_k$, where ΔM_{k+1} is the k+1 step transformation matrix defined with respect to a previous transformation matrix M_k; and M_e is the external transformation matrix defined by step (a) said external transformation matrix containing only rotations.

11. An interactive videographics system for animating a 3-D object, said system including:

at least one interactive user control input defining an external transformation matrix;

a multiplier that multiplies said external transformation matrix by a current transformation matrix and its inverse to produce a resultant matrix; and

a display coupled to said multiplier, said display displaying said object based at least in part on said resultant matrix.

12. A system as in claim 11 wherein said 3-D object is defined in an object modelling space having an origin, and said multiplier also multiplies said resultant matrix by a next animation frame transformation matrix constrained to specify translation only relative to said object modelling space origin.

13. A system as in claim 12 wherein said object includes at least one movable portion that is movable relative to a reference portion, and said system generates a connection matrix by concatenating a next animation frame transformation matrix with said external matrix for said movable portion relative to said object coordinate system origin.

14. A system as in claim 11 wherein said display object is modeled as a hierarchical system of articulated parts, and said multiplication is performed for at least some of said parts.

15. A system as in claim 11 wherein said object is modeled as an articulated system having a root defined by a reference object, and said multiplication is performed for said reference object.

16. A system as in claim 11 wherein said multiplier performs a multiplication for each step of a multi-step animation.

17. A system as in claim 11 wherein said system continuously concatenates the external transformation with a next animation step to produce a connection matrix.

18. A system as in claim 11further including concatenating the resultant with a connection matrix saved from a just previous animation step.

19. A system as in claim 11 wherein said external transformation matrix contains only rotations.

20. A system as in claim 11 wherein multiplier performs the following multiplication:

$$M''_{k+1} = M_{k+1}(-M^{t}_{k})\, M^{r}_{e}\, M^{t}_{k}\, M_{k}$$

where M_k and M_{k+1} are absolute transformation matrices of animation steps, k and k+1 defined with respect to the origin of the object's coordinate system; $M_{k+1} = \Delta M_{k+1} M_k$, where ΔM_{k+1} is the k+1 step transformation matrix defined with respect to a previous transformation matrix M_k; and M_e is the external transformation matrix defined by said external transformation matrix.

21. A method of authoring a three-dimensional animation sequence comprising:

(a) defining a three-dimensional object in terms of a reference object within a three-dimensional object coordinate system having an origin; and

(b) defining at least one sequence of animation matrices relating to said object, each of said animation matrices in said sequence being constrained to define a translation only, relative to said object coordinate system origin; and

(c) representing said three-dimensional object by a hierarchical data structure defining articulated segments of said object and said sequence of animation matrices.

22. A method as in claim 21 wherein said object is modeled as a hierarchical system of articulated segments, and said step (b) defines an animation of at least one of said segments relative to said reference object.

23. A method as in claim 21 wherein said object is modeled as an articulated system having a root defined by said reference object.

24. A method as in claim 21 wherein said sequence of animation matrices are adapted to be continuously concatenated with an external transformation matrix based on real-time user input.

25. A method as in claim 24 wherein said external transformation matrix contains only rotations.

26. A method as in claim 21 further including the step of developing and/or supplying, through use of said authoring system, a calculating routine for calculating

$$M''_{k+1} = M_{k+1}(-M^t_k)\, M^r_e\, M^t_k\, M_k$$

where M_k and M_{k+1} are absolute transformation matrices of animation steps, k and k+1 defined with respect to the origin of the object's coordinate system;

$M_{k+1} = \Delta M_{k+1} M_k$, where ΔM_{k+1} is the k+1 step transformation matrix defined with respect to a previous transformation matrix M_k; and M_e is the external transformation matrix defined by at least one external transformation matrix.

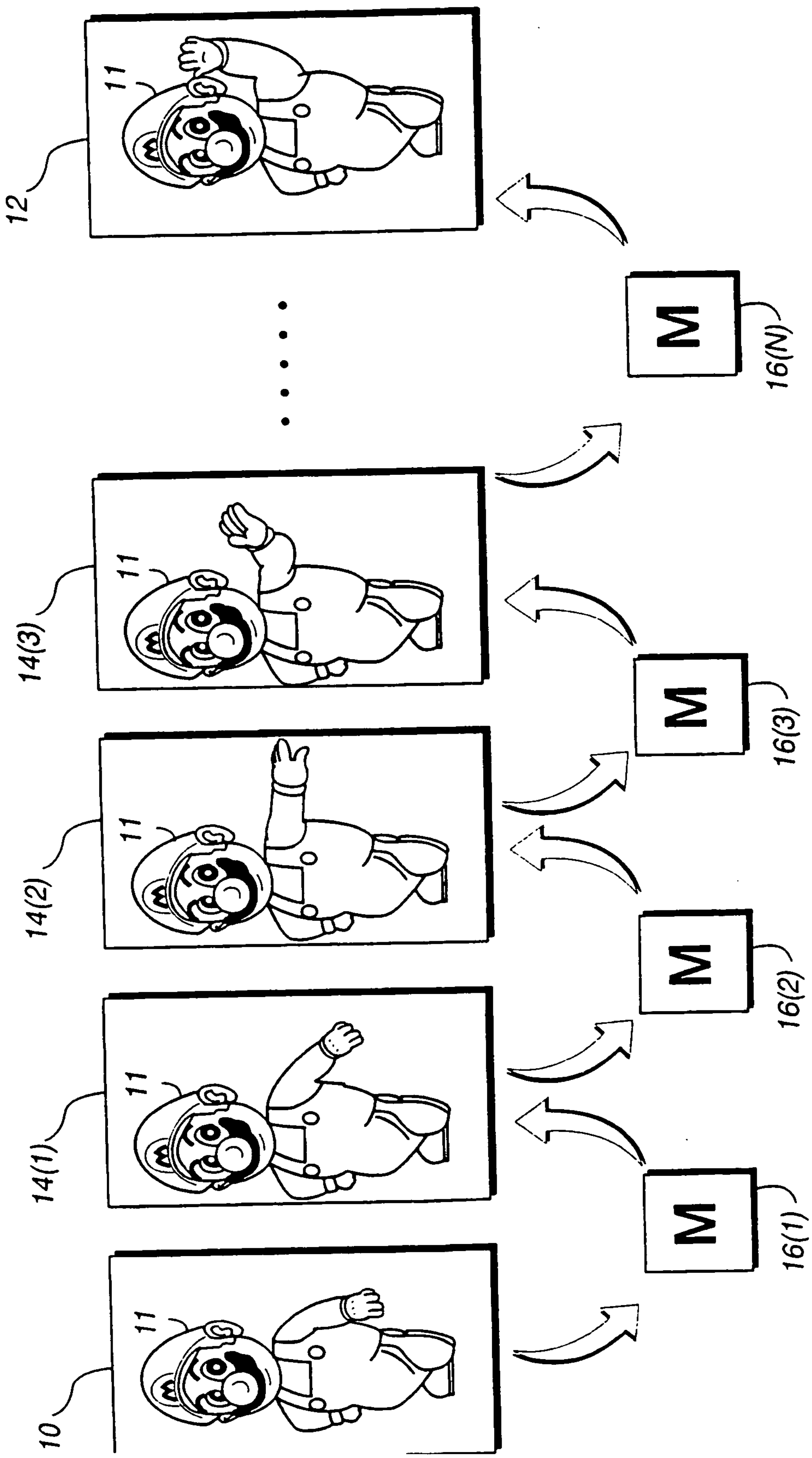
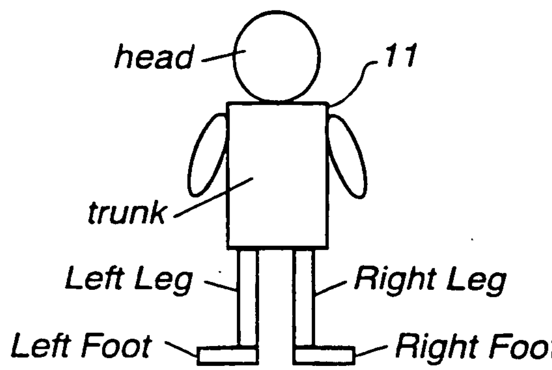


Fig. 1 (PRIOR ART)

Fig. 1A (PRIOR ART)



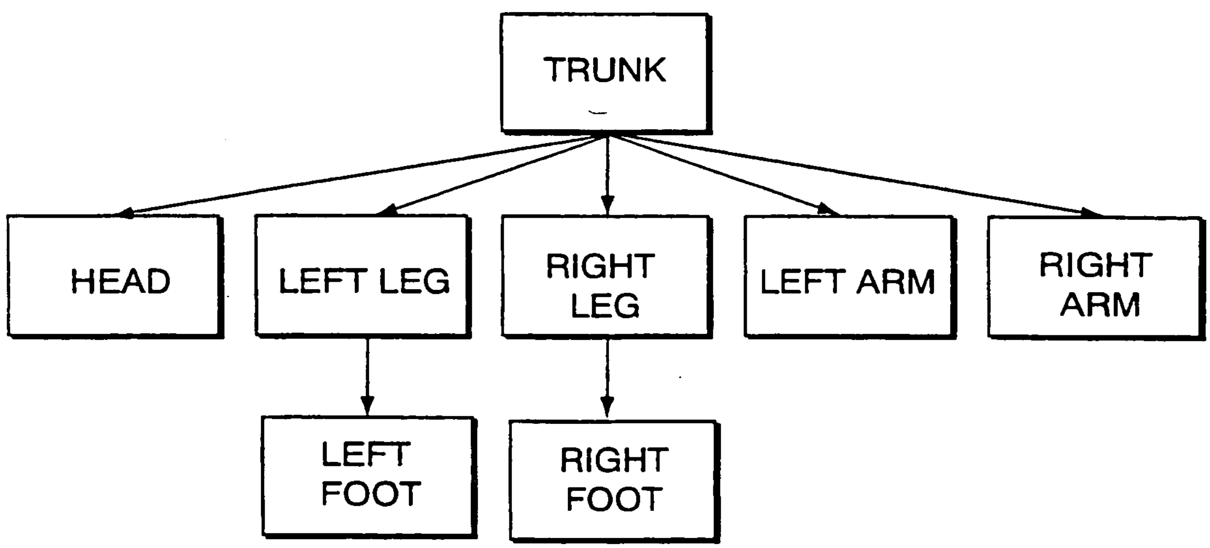


Fig. 1B (PRIOR ART)

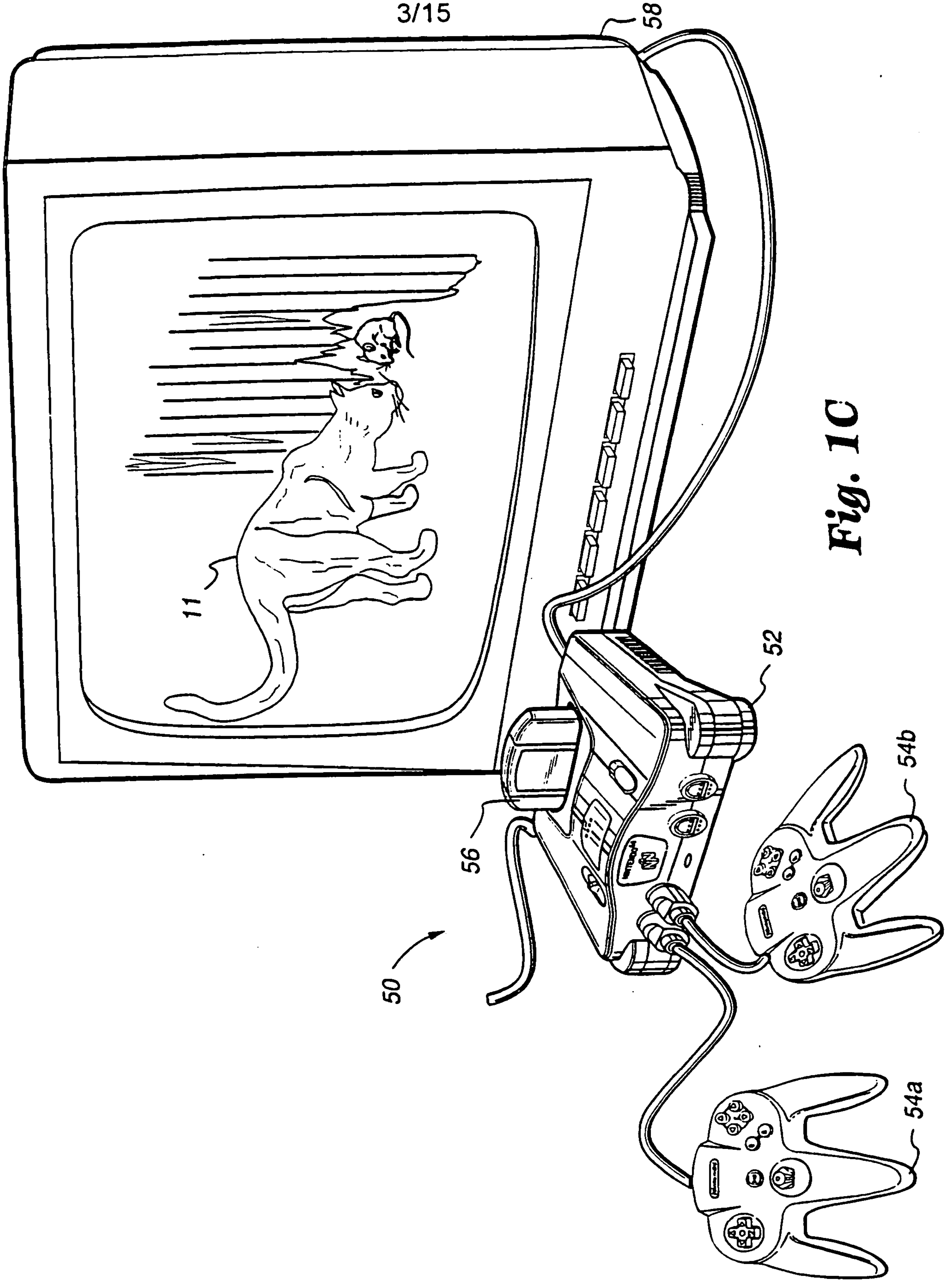


Fig. 1C

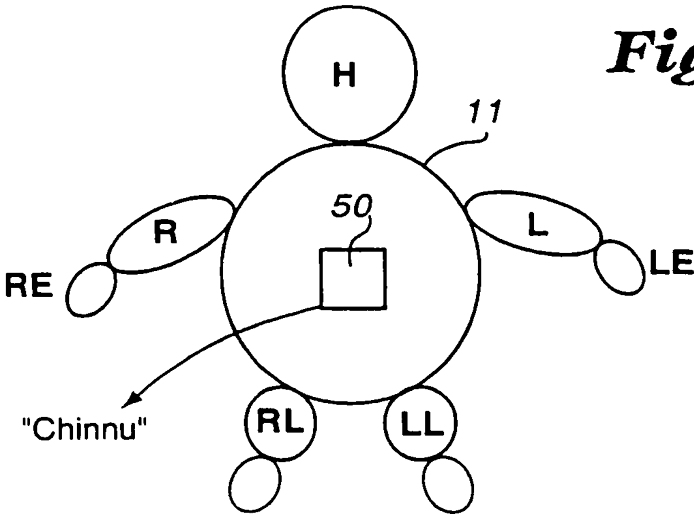


Fig. 2

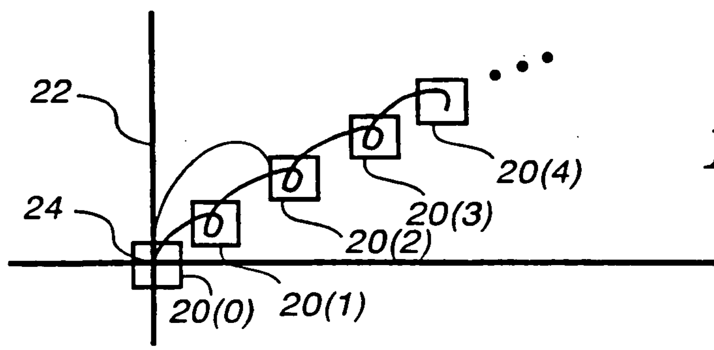


Fig. 2A

Fig. 2B

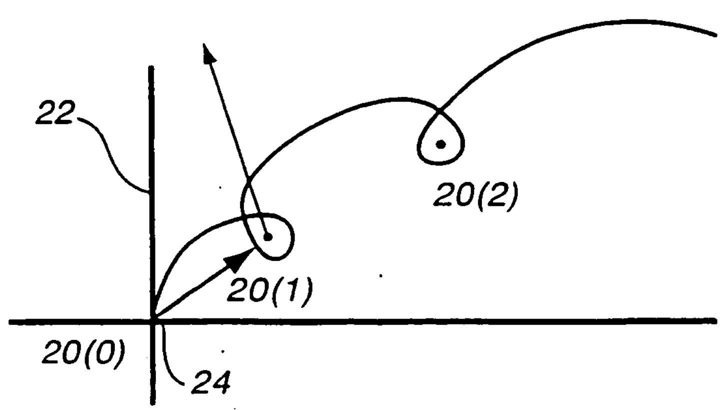
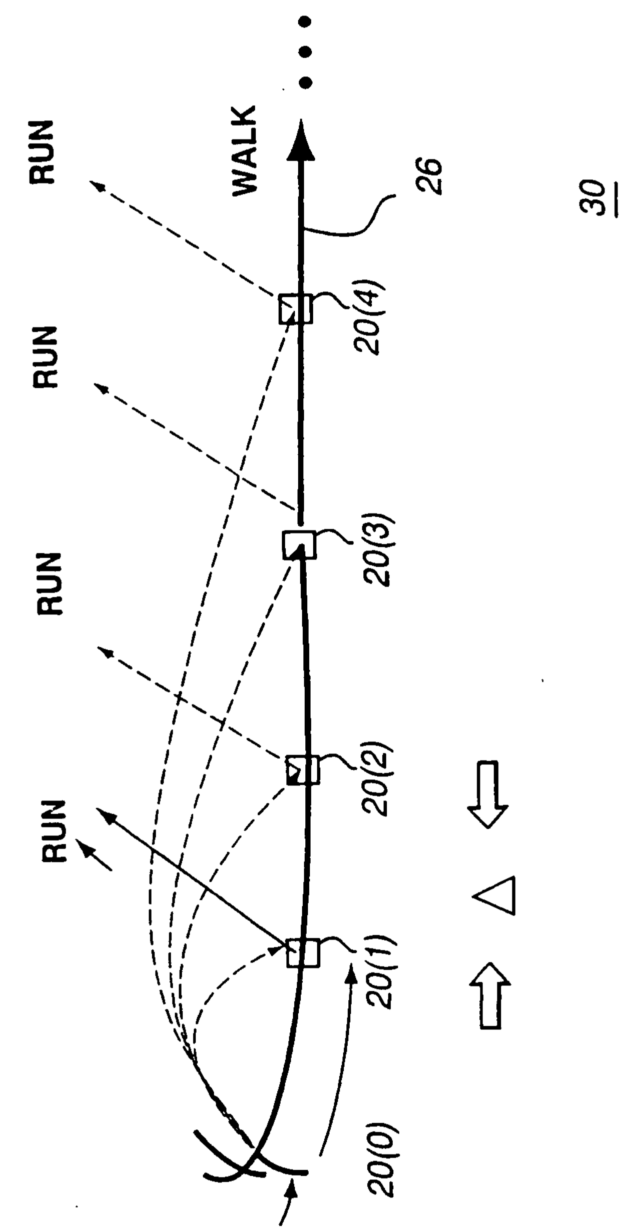


Fig. 2C



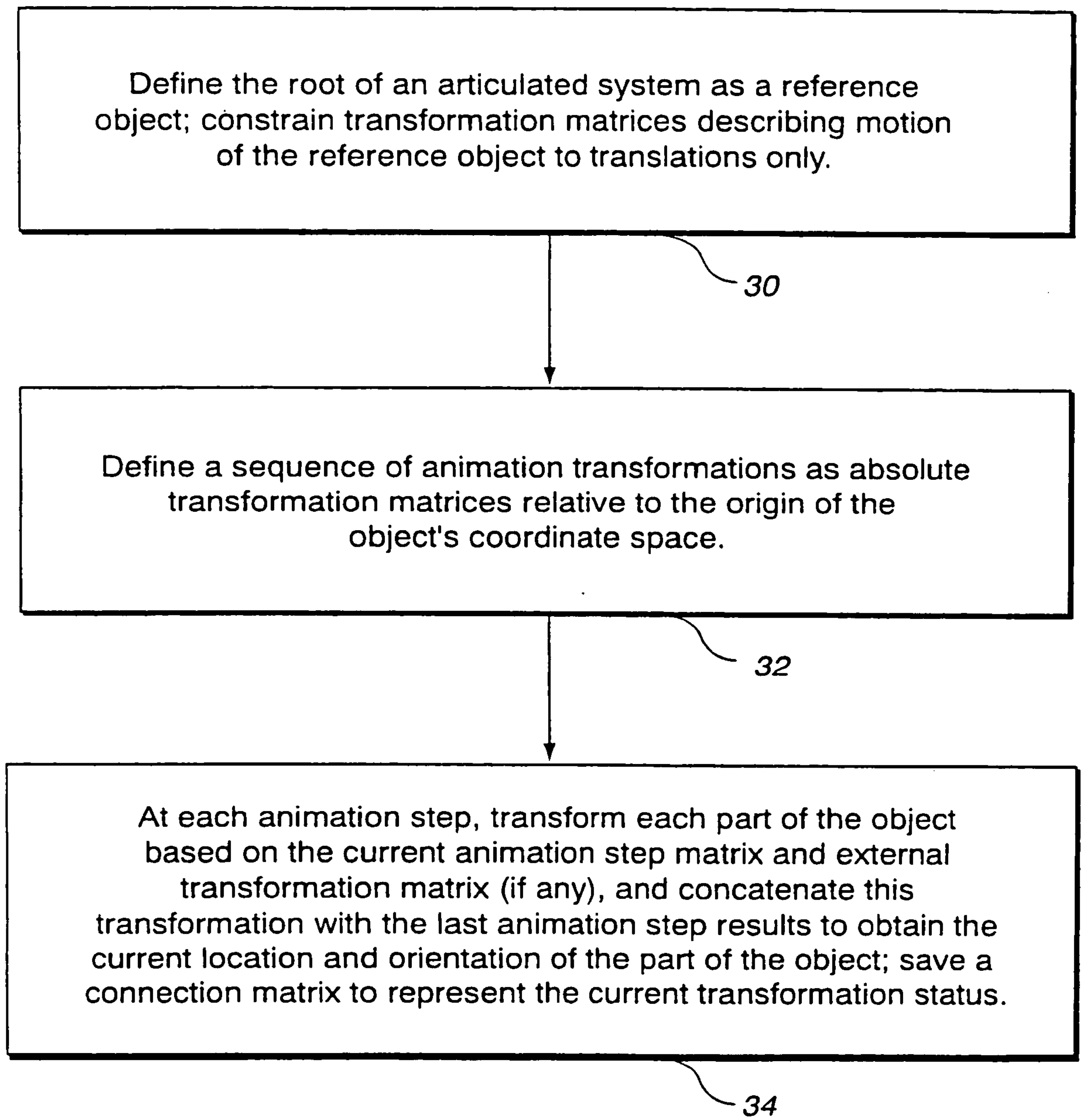

Define the root of an articulated system as a reference object; constrain transformation matrices describing motion of the reference object to translations only.
30
Define a sequence of animation transformations as absolute transformation matrices relative to the origin of the object's coordinate space.
32
At each animation step, transform each part of the object based on the current animation step matrix and external transformation matrix (if any), and concatenate this transformation with the last animation step results to obtain the current location and orientation of the part of the object; save a connection matrix to represent the current transformation status.
34

Fig. 3

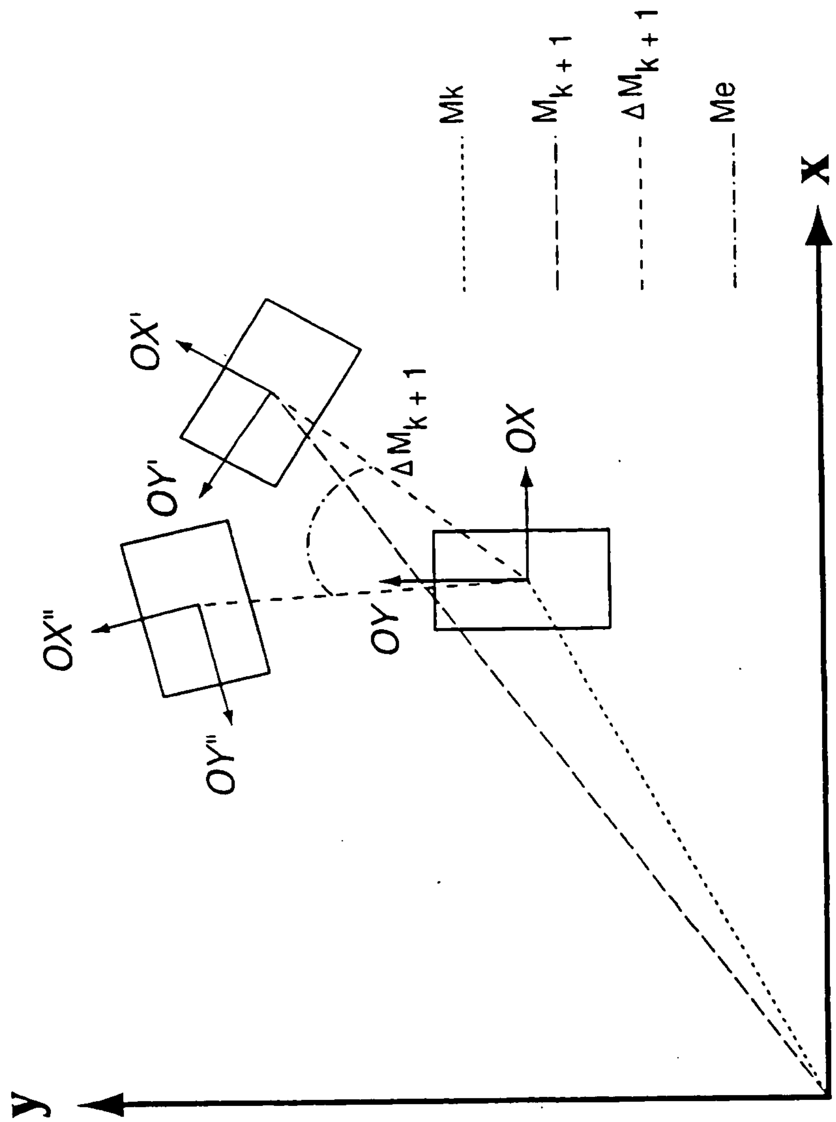


Fig. 3A

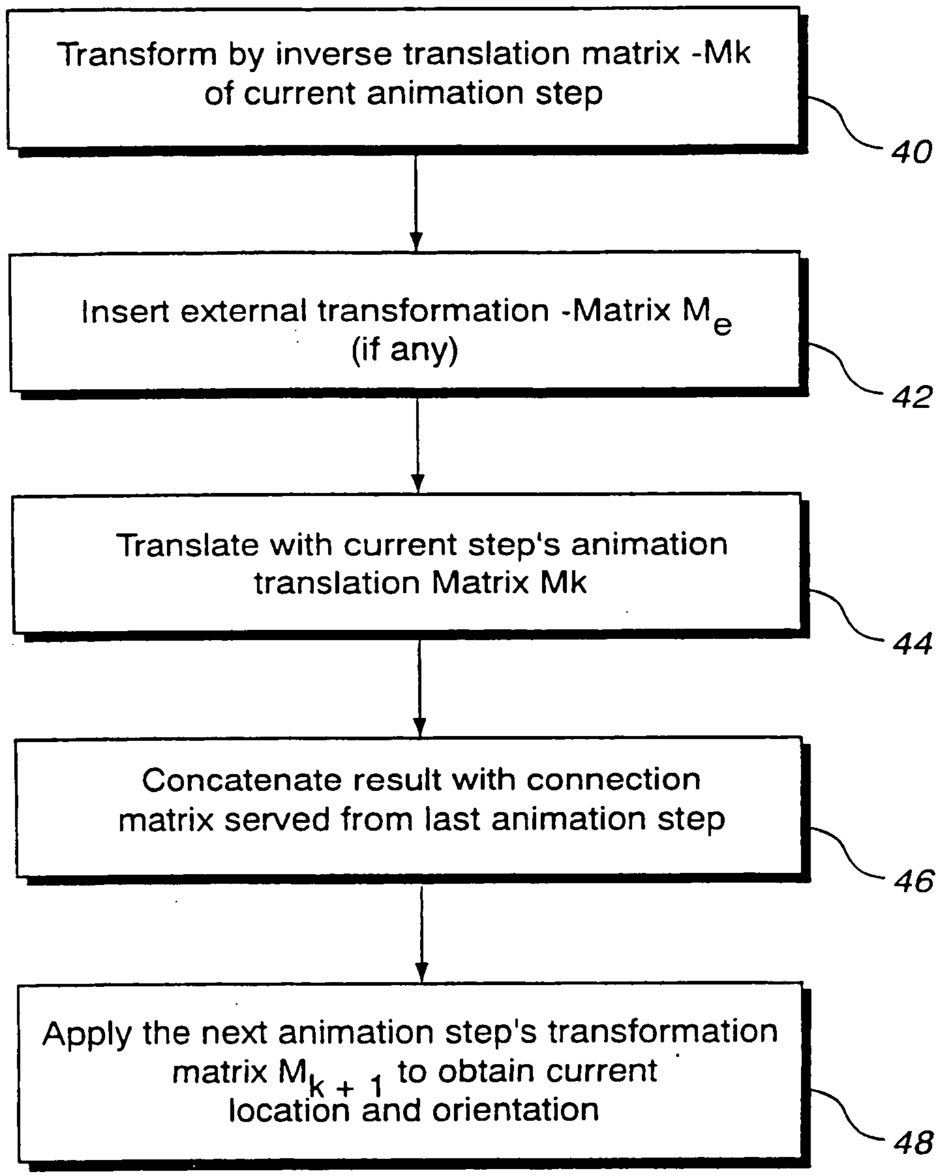

Transform by inverse translation matrix -Mk of current animation step
40
Insert external transformation -Matrix M_e (if any)
42
Translate with current step's animation translation Matrix Mk
44
Concatenate result with connection matrix served from last animation step
46
Apply the next animation step's transformation matrix M_{k+1} to obtain current location and orientation
48

Fig. 4

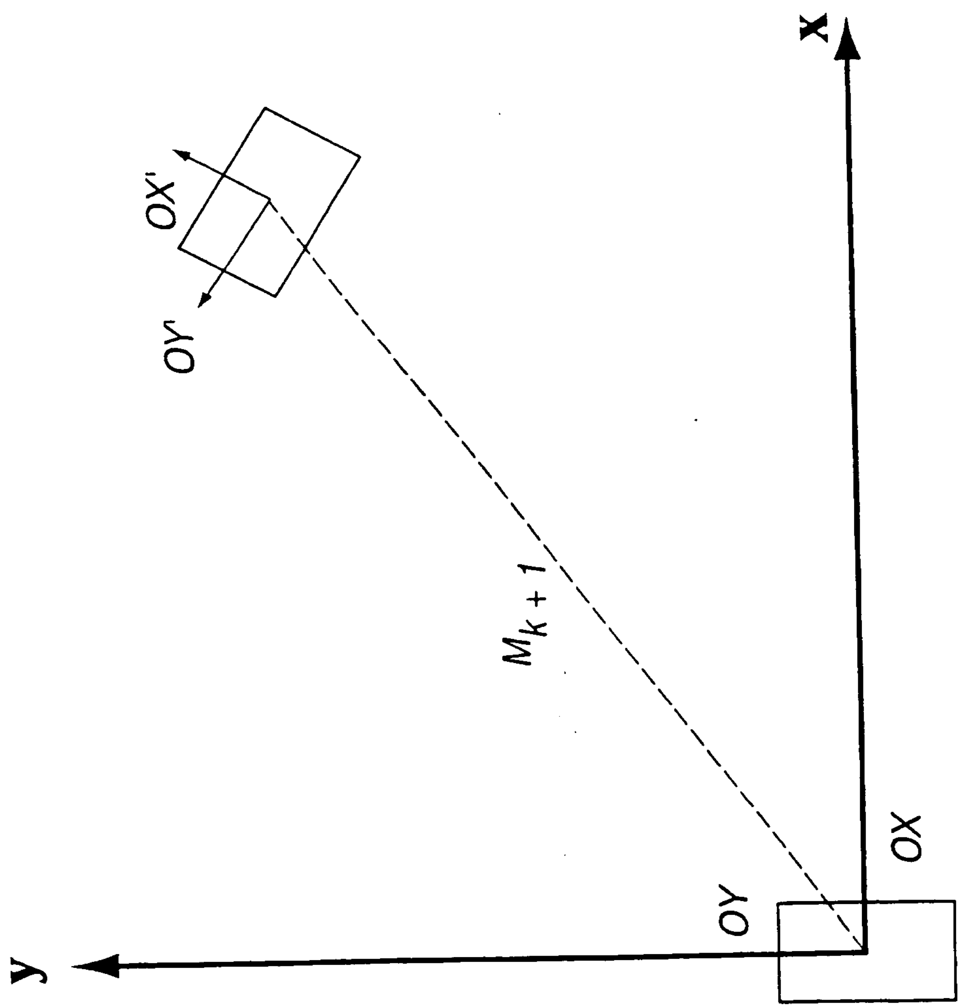


Fig. 4A Object is transformed by transformation matrix M_{k+1}

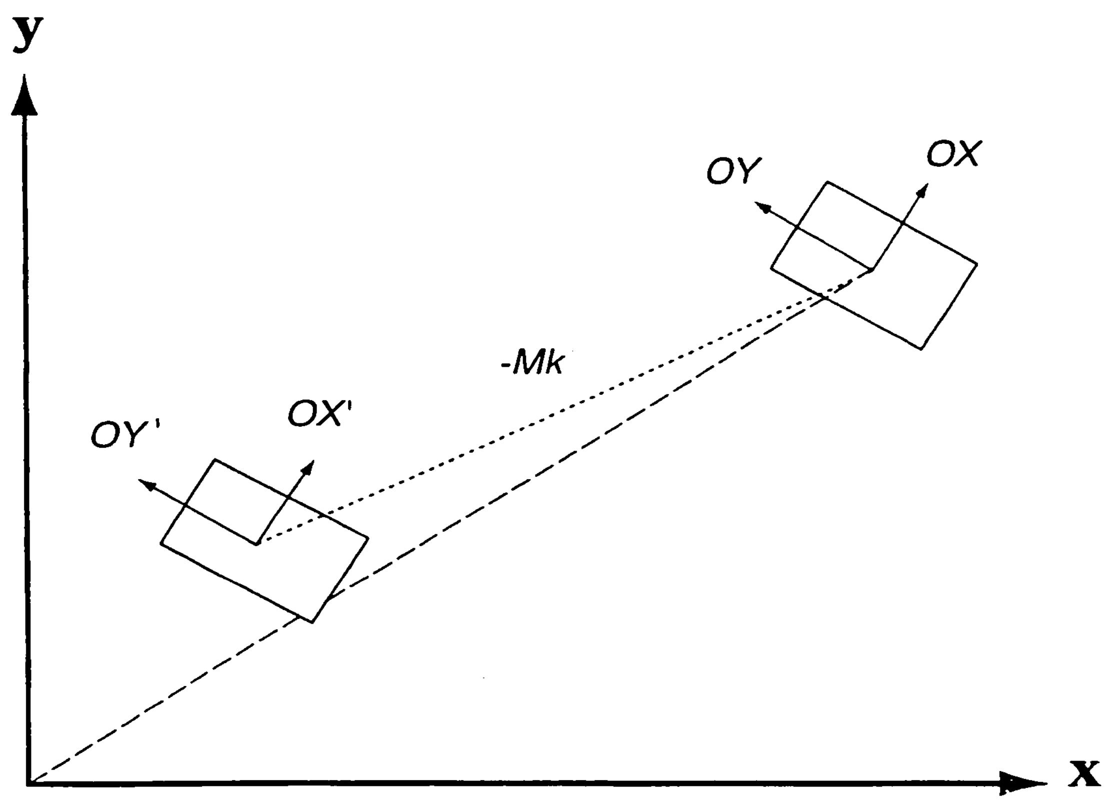


Fig. 4B *Object is translated by translation matrix* $-M_k$

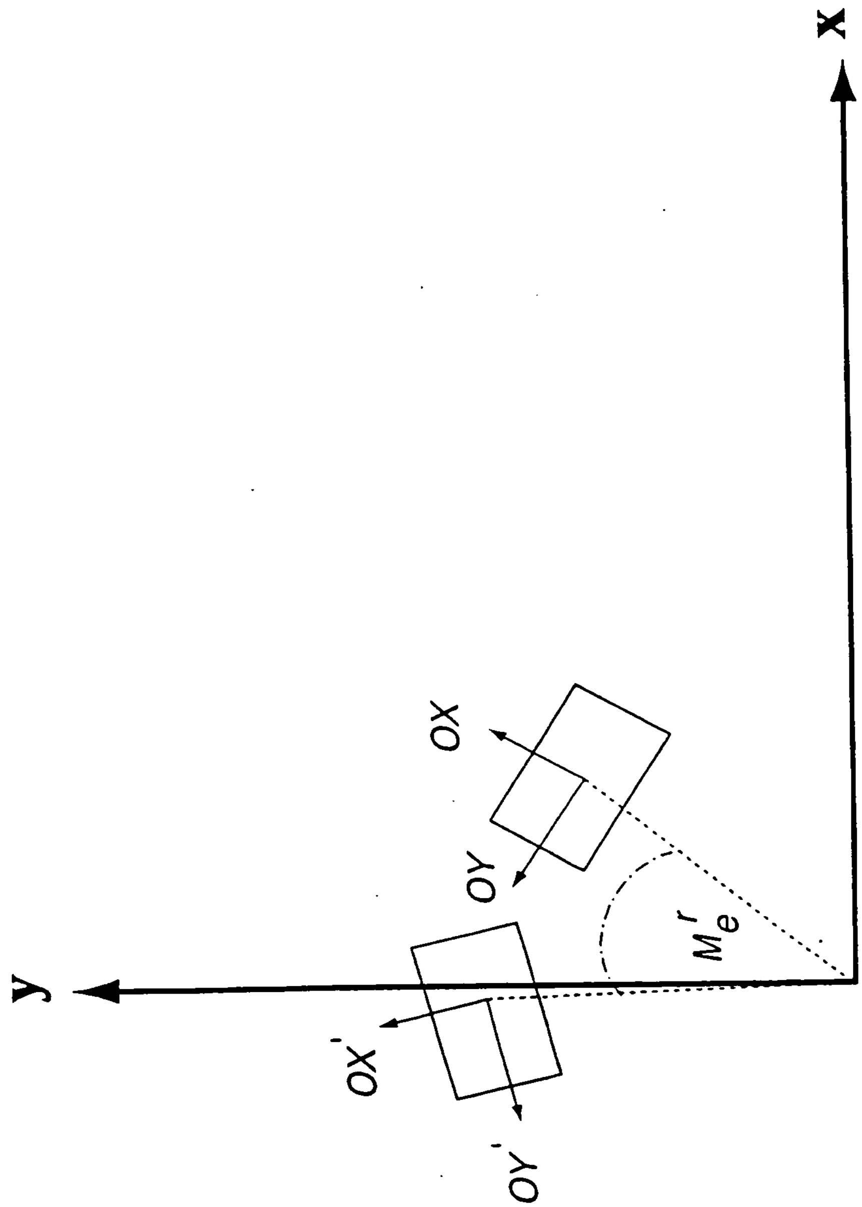


Fig. 4C Object is transformed by external rotation matrix M_e

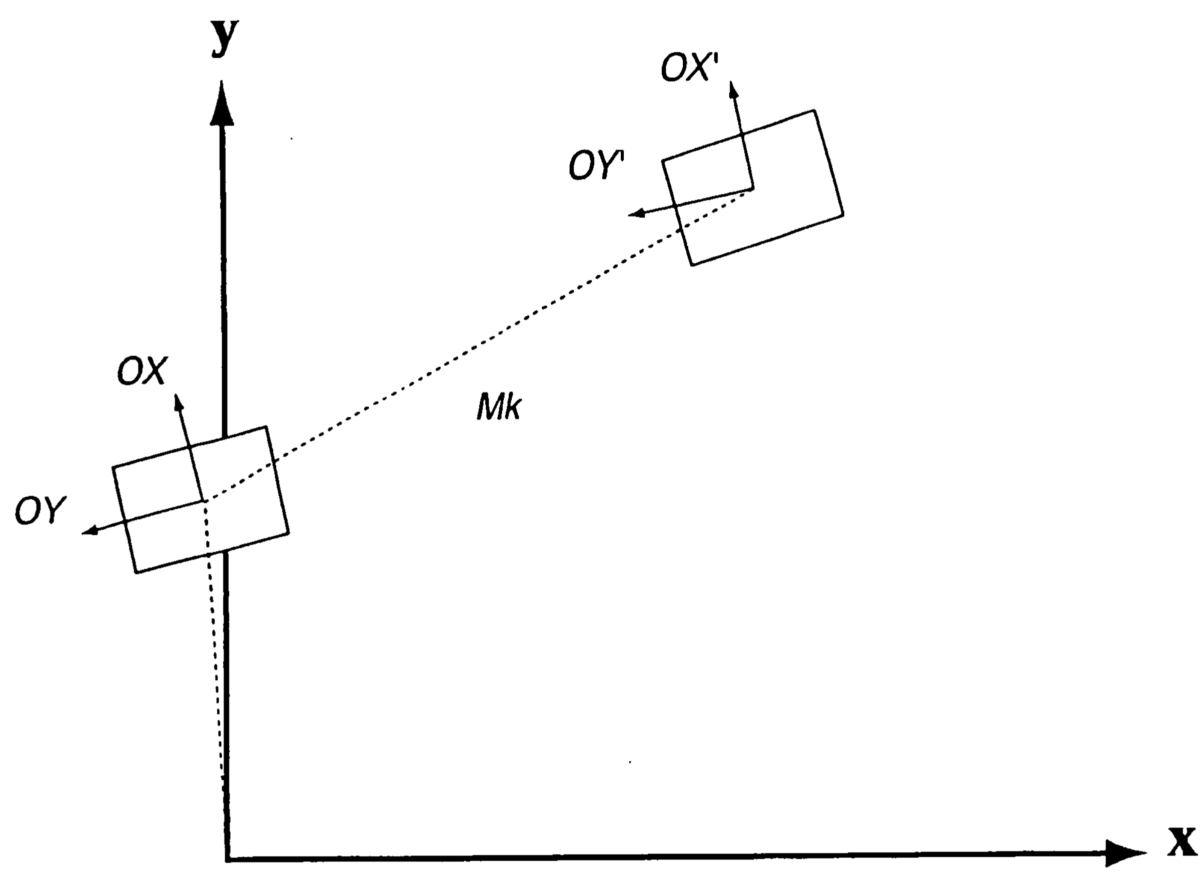


Fig. 4D *Object is translated by translation matrix M_k*

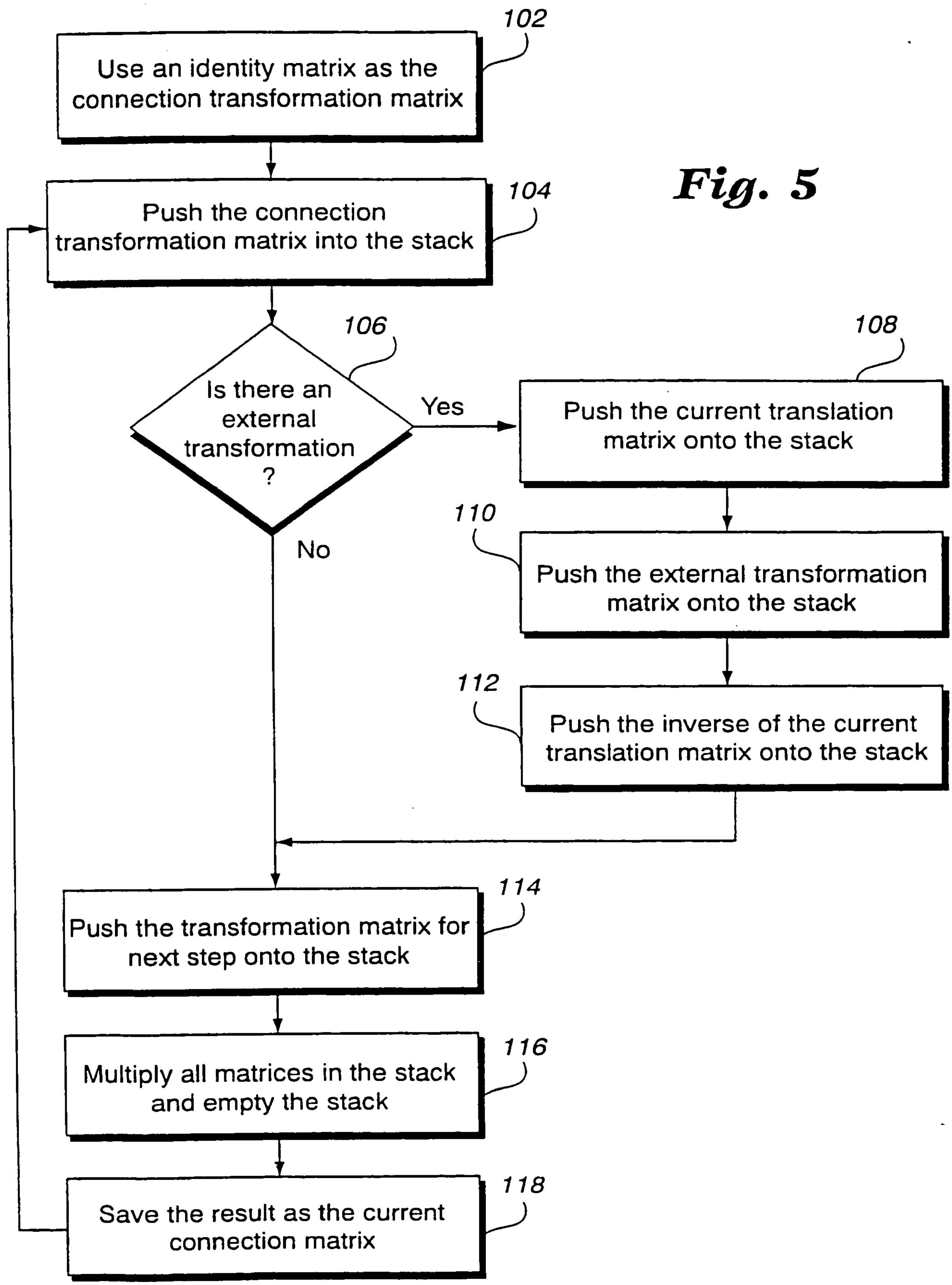
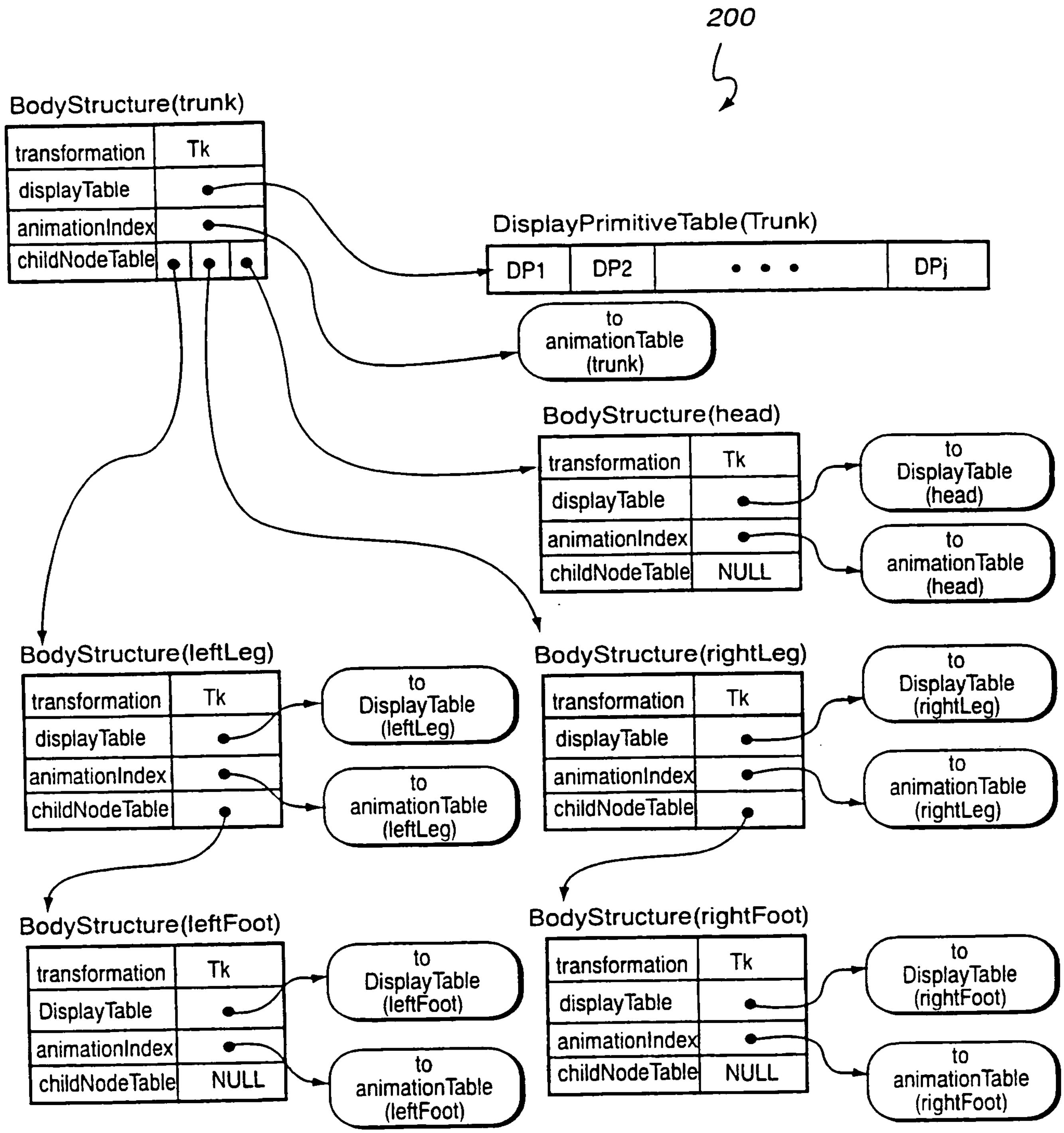

102
Use an identity matrix as the connection transformation matrix
104
Push the connection transformation matrix into the stack
106
Is there an external transformation ?
Yes
No
108
Push the current translation matrix onto the stack
110
Push the external transformation matrix onto the stack
112
Push the inverse of the current translation matrix onto the stack
114
Push the transformation matrix for next step onto the stack
116
Multiply all matrices in the stack and empty the stack
118
Save the result as the current connection matrix

Fig. 5



200
BodyStructure(trunk)
transformation
Tk
displayTable
animationIndex
childNodeTable
DisplayPrimitiveTable(Trunk)
DP1
DP2
• • •
DPj
to animationTable (trunk)
BodyStructure(head)
transformation
Tk
displayTable
animationIndex
childNodeTable
NULL
to DisplayTable (head)
to animationTable (head)
BodyStructure(leftLeg)
transformation
Tk
displayTable
animationIndex
childNodeTable
to DisplayTable (leftLeg)
to animationTable (leftLeg)
BodyStructure(rightLeg)
transformation
Tk
displayTable
animationIndex
childNodeTable
to DisplayTable (rightLeg)
to animationTable (rightLeg)
BodyStructure(leftFoot)
transformation
Tk
DisplayTable
animationIndex
childNodeTable
NULL
to DisplayTable (leftFoot)
to animationTable (leftFoot)
BodyStructure(rightFoot)
transformation
Tk
displayTable
animationIndex
childNodeTable
NULL
to DisplayTable (rightFoot)
to animationTable (rightFoot)

Fig. 6A

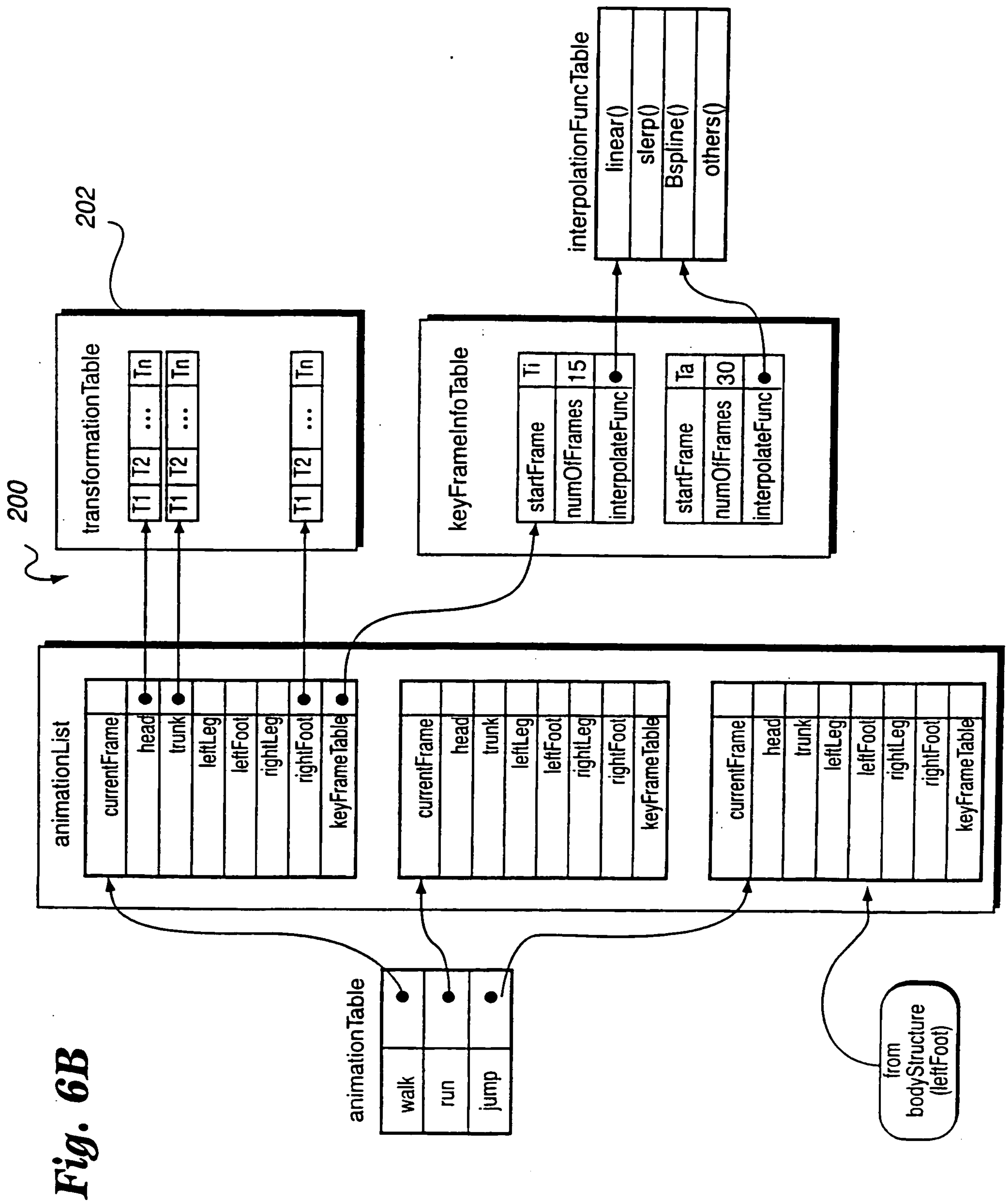

200
202
transformationTable
T1
T2
...
Tn
keyFrameInfoTable
startFrame
Ti
numOfFrames
15
interpolateFunc
Ta
30
interpolationFuncTable
linear()
slerp()
Bspline()
others()
animationList
currentFrame
head
trunk
leftLeg
leftFoot
rightLeg
rightFoot
keyFrameTable
animationTable
walk
run
jump
from bodyStructure (leftFoot)

Fig. 6B

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/27696

A. CLASSIFICATION OF SUBJECT MATTER
IPC(7) :G06T 15/70
US CL : 345/473, 474, 475, 427, 437, 438 and 439.
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 345/473, 474, 475, 427, 437, 438 and 439.

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EAST, WEST, IEEE
animation, transformation matrix, articulate, hierarchy, rotation orientation, keyframe

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 4,600,919 A (STERN) 15 July 1986, col. 2, lines 17-27; col. 4, lines 45-50; col. 9, lines 2-8; col. 8, lines 22-31; col. 6, line 7-25; col. 9, lines 14-22; col. 2, lines 36-45; col. 9, lines 19-25; col. 9, lines 14-25; col. 9, lines 14-20; col. 6, lines 7-12; col. 8, lines 22-60; col. 5, line 63 through col. 6, line 25.	1-4, 6-14, 16-22, 24-26
Y	US 4,797,836 A (WITEK et al.) 10 January 1989, col. 4, lines 36-47; col. 2, lines 40-43; col. 1, lines 7-13.	1-4, 6-14, 16-22, 24-26
X,E	US 6,088,042 A (HANDELMAN et al.) 11 July 2000, col. 6, line 50 through col. 7, line 5).	5, 15, 23

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 NOVEMBER 2000	28 NOV 2000

Name and mailing address of the ISA/US	Authorized officer
Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	KIMBINH NGUYEN Eugenia Logan
Facsimile No. (703) 305-3230	Telephone No. (703) 305-9683

Form PCT/ISA/210 (second sheet) (July 1998) ★